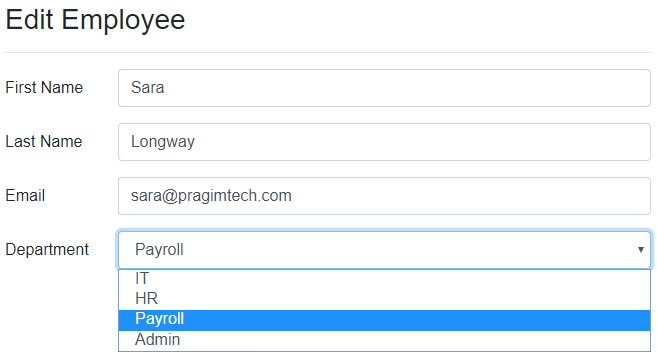
Binding select element with database data in Blazor

In this video we will discuss how to bind a select element with database data in Blazor. When the **Edit Employee** form loads, we want to retrieve the list of all departments from the database and bind them to the Department dropdownlist. In addition to binding the list of all departments, we also want the employee's department to be selected.

For example, if the employee's department is Payroll, in addition to displaying the list of all departments, we want the Payroll department to be selected.



Blazor input select example

<InputSelect id="department" @bind-Value="DepartmentId">

@foreach (var dept in Departments)

{

<option value="@dept.DepartmentId">@dept.DepartmentName</option>

}

</InputSelect>

Copy

Code explanation

* We use the InputSelect component, to render an html select element
* Departments property in the component class carries the list of all departments to the view.
* The foreach loop, loops through the list and creates a select element option for each department.
* The value for the option is the department id and the display text is the department name.
* If the employee belongs to IT department, it should be selected. For this we use @bind-Value attribute. This attribute provides two-way data-binding i.e on the initial page load, the employee department is selected and if we change the selection, the newly selected department value is automatically passed to the DepartmentId property in the component class.
* Binding the select element to an integer is not supported and throws the following exception.  
  Microsoft.AspNetCore.Components.Forms.InputSelect`1[System.Int32] does not support the type 'System.Int32'

Edit Employee Component View (EditEmployee.razor)

@page "/editemployee/{id}"

@inherits EditEmployeeBase

<EditForm Model="@Employee">

<h3>Edit Employee</h3>

<hr />

<div class="form-group row">

<label for="firstName" class="col-sm-2 col-form-label">

First Name

</label>

<div class="col-sm-10">

<InputText id="firstName" class="form-control" placeholder="First Name"

@bind-Value="Employee.FirstName" />

</div>

</div>

<div class="form-group row">

<label for="lastName" class="col-sm-2 col-form-label">

Last Name

</label>

<div class="col-sm-10">

<InputText id="lastName" class="form-control" placeholder="Last Name"

@bind-Value="Employee.LastName" />

</div>

</div>

<div class="form-group row">

<label for="email" class="col-sm-2 col-form-label">

Email

</label>

<div class="col-sm-10">

<InputText id="email" class="form-control" placeholder="Email"

@bind-Value="Employee.Email" />

</div>

</div>

<div class="form-group row">

<label for="department" class="col-sm-2 col-form-label">

Department

</label>

<div class="col-sm-10">

<InputSelect id="department" @bind-Value="DepartmentId" class="form-control">

@foreach (var dept in Departments)

{

<option value="@dept.DepartmentId">@dept.DepartmentName</option>

}

</InputSelect>

</div>

</div>

</EditForm>

Copy

Edit Employee Component Class (EditEmployeeBase.cs)

using EmployeeManagement.Models;

using EmployeeManagement.Web.Services;

using Microsoft.AspNetCore.Components;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

namespace EmployeeManagement.Web.Pages

{

public class EditEmployeeBase : ComponentBase

{

[Inject]

public IEmployeeService EmployeeService { get; set; }

public Employee Employee { get; set; } = new Employee();

[Inject]

public IDepartmentService DepartmentService { get; set; }

public List<Department> Departments { get; set; } = new List<Department>();

public string DepartmentId { get; set; }

[Parameter]

public string Id { get; set; }

protected async override Task OnInitializedAsync()

{

Employee = await EmployeeService.GetEmployee(int.Parse(Id));

Departments = (await DepartmentService.GetDepartments()).ToList();

DepartmentId = Employee.DepartmentId.ToString();

}

}

}

Copy

The rest of the code in this article, explains how to retrieve Departments data from the database table using a REST API.

REST API Project - IDepartmentRepository Interface

These 2 methods perform database operations and we want them to be executed asynchronously so they return a task.

public interface IDepartmentRepository

{

Task<IEnumerable<Department>> GetDepartments();

Task<Department> GetDepartment(int departmentId);

}

Copy

REST API Project - DepartmentRepository Class

public class DepartmentRepository : IDepartmentRepository

{

private readonly AppDbContext appDbContext;

public DepartmentRepository(AppDbContext appDbContext)

{

this.appDbContext = appDbContext;

}

public async Task<Department> GetDepartment(int departmentId)

{

return await appDbContext.Departments

.FirstOrDefaultAsync(d => d.DepartmentId == departmentId);

}

public async Task<IEnumerable<Department>> GetDepartments()

{

return await appDbContext.Departments.ToListAsync();

}

}

Copy

REST API Project - Departments REST API Controller

[Route("api/[controller]")]

[ApiController]

public class DepartmentsController : ControllerBase

{

private readonly IDepartmentRepository departmentRepository;

public DepartmentsController(IDepartmentRepository departmentRepository)

{

this.departmentRepository = departmentRepository;

}

[HttpGet]

public async Task<ActionResult> GetDepartments()

{

try

{

return Ok(await departmentRepository.GetDepartments());

}

catch (Exception)

{

return StatusCode(StatusCodes.Status500InternalServerError,

"Error retrieving data from the database");

}

}

[HttpGet("{id:int}")]

public async Task<ActionResult<Department>> GetDepartment(int id)

{

try

{

var result = await departmentRepository.GetDepartment(id);

if (result == null)

{

return NotFound();

}

return result;

}

catch (Exception)

{

return StatusCode(StatusCodes.Status500InternalServerError,

"Error retrieving data from the database");

}

}

}

Copy

Blazor Web Project - IDepartmentService Interface

public interface IDepartmentService

{

Task<IEnumerable<Department>> GetDepartments();

Task<Department> GetDepartment(int id);

}

Copy

Blazor Web Project - DepartmentService Class

public class DepartmentService : IDepartmentService

{

private readonly HttpClient httpClient;

public DepartmentService(HttpClient httpClient)

{

this.httpClient = httpClient;

}

public async Task<Department> GetDepartment(int id)

{

return await httpClient.GetJsonAsync<Department>($"api/departments/{id}");

}

public async Task<IEnumerable<Department>> GetDepartments()

{

return await httpClient.GetJsonAsync<Department[]>("api/departments");

}

}

Copy

Blazor Web Project - Startup.cs

In ConfigureServices method of the Startup class register HttpClient Services using AddHttpClient method.

public void ConfigureServices(IServiceCollection services)

{

services.AddRazorPages();

services.AddServerSideBlazor();

services.AddHttpClient<IEmployeeService, EmployeeService>(client =>

{

client.BaseAddress = new Uri("https://localhost:44379/");

});

services.AddHttpClient<IDepartmentService, DepartmentService>(client =>

{

client.BaseAddress = new Uri("https://localhost:44379/");

});

}